OASIS Symposium: Reliable Infrastructures for XML
Critical Comparison of WS-RM and WS-R
April 27, 2004

Christopher Ferris
Senior Technical Staff Member
IBM Emerging Technology

Agenda

- Common features
- What's different?
- Conclusions
- Resources

Common features

- **Both are SOAP-based protocol extensions**
- **Both provide same delivery assurance semantics**
  - At-Least-Once
  - At-Most-Once
  - Exactly-Once
  - Ordered delivery
- **Each message has a group identifier and unique sequence number**
  - WS-R has MessageId
  - WS-RM has Sequence
  - Both are roughly equivalent
- **Both support "piggy-backing" for bi-directional reliability**

Common features (continued)

- **Each has some form of acknowledgement**
  - Each provides for a range of messages to be acknowledged
- **Each can survive loss of acknowledgement messages**
  - Although it isn't abundantly clear that there is any *guarantee* of survival on part of WS-R
  - WS-R appears to have no requirement that *all* messages in a group be acknowledged in each SequenceReplies:

WS-R - "This element MUST contain the values of the original MessageIds of the messages delivered for a group, and for each Fault Code being reported, the MessageIds of messages which encountered the particular Fault Code."

WS-RM - "Every acknowledgement issued by the RM Destination MUST include within an acknowledgement range or ranges the sequence number of every message successfully received by the RM Destination and MUST exclude sequence numbers of any messages not yet received."

What's different

- Complexity levels
- Processing model
- Nack
- Composability
  - Addressing, Policy
- Implementation details
- Efficiency of resource reclamation

Complexity

- **WS-R is 78 pages as compared with WS-RM's 40**
- **WS-R has 62 occurances of the word "if", WS-RM only 22**
  - Much of the conditional language in the WS-R spec relates to implementation detail which should be out of scope of a protocol specification
- **WS-R has *two* versions of its schema; one for SOAP1.1 and another for SOAP1.2**
  - Why? This means that there are effectively two versions of the protocol just to accommodate the soap:mustUnderstand attribute?
- **WS-R seems to have invented its own fault mechanism rather than building on SOAP Fault**
- **It is unclear that WS-R offers any guarantee of correctness**

Example WS-R Request Header

```
<Request
 xmlns="http://www.oasis-open.org/committees/wsrm/schema/1.1/SOAP1.1"
 xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
 xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
 soap:mustUnderstand="1">
 <MessageId groupId="mid://20040202.103832@oasis-open.org/">
   <SequenceNum number="0"
     groupExpiryTime="2005-02-02T03:00:33-31:00" />
 </MessageId>
 <ExpiryTime>2004-09-07T03:01:03-03:50</ExpiryTime>
 <ReplyPattern>Response</ReplyPattern>
 <AckRequested/>
 <DuplicateElimination/>
 <MessageOrder/>
</Request>
```

Example WS-RM Sequence Header

```
<Sequence xmlns="http://schemas.xmlsoap.org/ws/2004/03/rm"
  xmlns:wsu="http://schemas.xmlsoap.org/ws/2002/07/utility">
  <wsu:Identifier>http://fabrikam123.com/abc</wsu:Identifier>
  <MessageNumber>10</MessageNumber>
  <LastMessage/>
</Sequence>
```

Example WS-R Response Header

```
<Response
  xmlns="http://www.oasis-open.org/committees/wsrm/schema/1.1/SOAP1.1"
  xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  soap:mustUnderstand="1" replyPattern=”Callback”>
  <NonSequenceReply groupId="mid://20040202.103832@oasis-open.org" />
  <NonSequenceReply groupId="mid://20040202.103811@oasis-open.org"
    fault=”wsrm:PermanentProcessingFailure” />
  <SequenceReplies groupId="mid://20040202.103807@oasis-open.org/">
    <ReplyRange from="1" to="4" />
    <ReplyRange from="5" to="5" fault=”wsrm:InvalidRequest” />
    <ReplyRange from="6" to="42" />
  </SequenceReplies>
</Response>
```

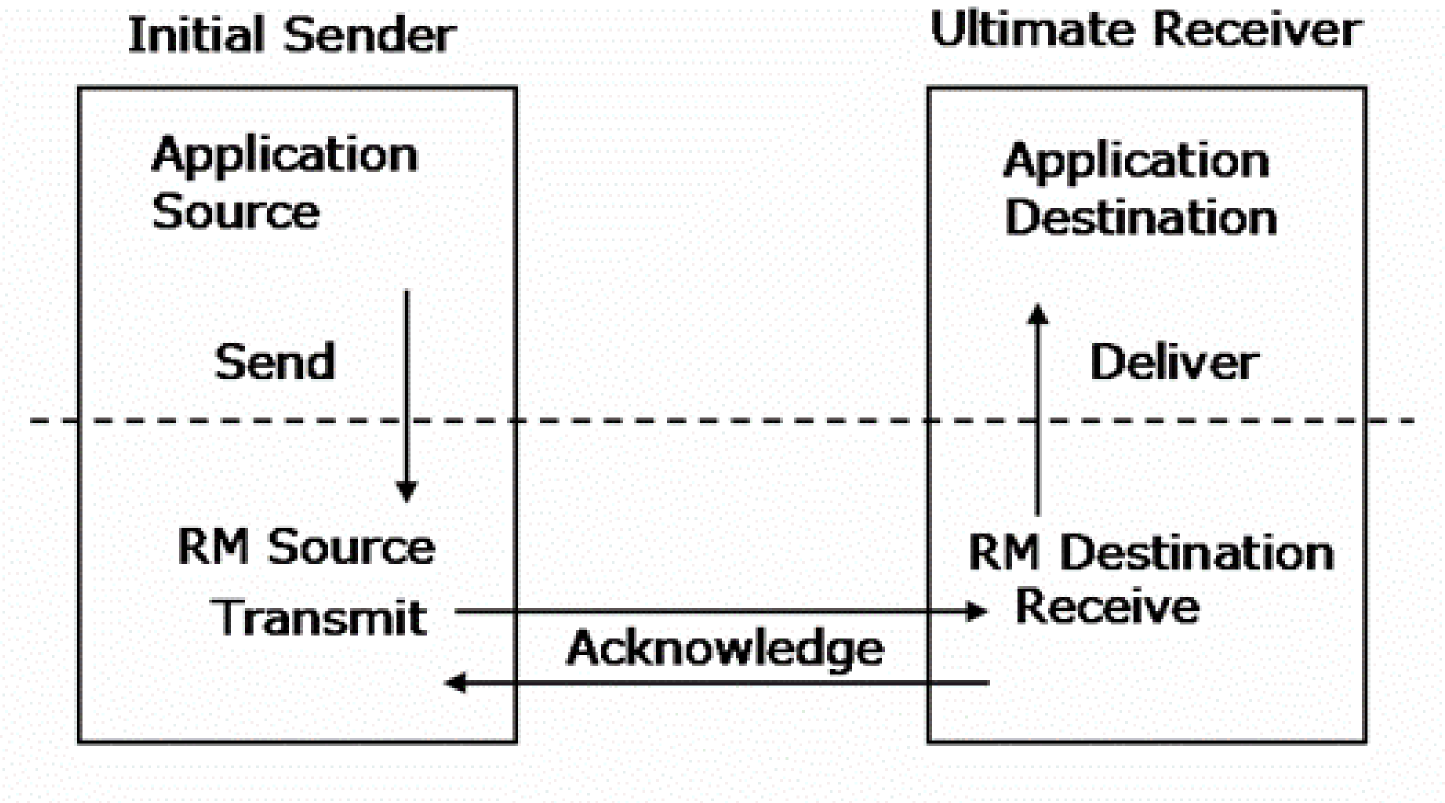
Example WS-RM SequenceAcknowledgement

```
<SequenceAcknowledgement
  xmlns="http://schemas.xmlsoap.org/ws/2004/03/rm"
  xmlns:wsu="http://schemas.xmlsoap.org/ws/2002/07/utility">
  <wsu:Identifier>http://fabrikam123.com/abc</wsu:Identifier>
  <AcknowledgementRange Upper="2" Lower="1"/>
  <AcknowledgementRange Upper="6" Lower="4"/>
  <AcknowledgementRange Upper="10" Lower="8"/>
</SequenceAcknowledgement>
```

Processing model

- **WS-R acknowledges messages only after they have been *delivered to the application***
  - *IMO, this completely misses the whole point*
  - *What if the application is off-line or busy processing other messages? Will the sending RMP keep resending just because the receiving RMP has not delivered messages to the application?*
- **WS-RM acknowledges *receipt* of a message**
  - Preserves separation of concerns; an application level acknowledgement needs to be an application-level message e.g. equivalent of an EDI 855 PO Acknowledgement
  - RM destination takes responsibility for delivery to application and acknowledges receipt immediately

WS-RM Processing Model



Nack

- **WS-RM added a Nack with the express purpose to allow an RM Destination to request that the RM Source resend a message**
- **WS-R provides for faults which could possibly be leveraged to this effect**
  - Not clear why a source needs to know why, only adds to complexity

Composability

- **WS-R Message Reply Patterns**
  - Sprinkles in addressing aspect that really should be orthogonal
  - Not at all clear that WS-R can effectively be composed with WS-Addressing
- **WS-RM designed for composability with other WS-* specs**
  - While WS-RM depends upon some form of addressing, the protocol itself is completely independent. Furthermore, WS-Addressing is not dependent upon WS-RM
  - Can be composed with WS-Policy*, WS-Security*, …

Implementation details

- **WS-R imposes requirements on implementation. e.g.**

  "NOTES: Given the above definition of ExpiryTime, in case Duplicate Elimination is required, when a received message is processed, it is sufficient to only check for its duplicates among MessageIds of past messages that have not expired yet at the time of the duplicate check."

  "Section 5 Operational Aspects and Semantics"

  – Implementation detail/advice has no place in a protocol spec

- **WS-RM defines the protocol exclusively in terms of what appears "on the wire"**

  – Provides more flexibility in potential use cases for WS-RM

Efficiency of resource reclamation

- **WS-R depends upon synchronization of clocks for resource reclamation**
- **WS-RM adds capability for destination to allocate Sequence identifiers**
  - CreateSequence, TerminateSequence operations defined
  - If RM Destination receives Sequence/Identifier that it does not recognize, then it can be assured that it can safely disregard the message
  - RM Destination can reclaim resources associated with a Sequence as soon as it receives TerminateSequence message
  - RM Destination can reclaim resources after Sequence expires even if it never receives the TerminateSequence message

Conclusions

- **Similar if not identical requirements**
- **Similar feature set**
- **Different approaches**
- **Time will tell which specification the market prefers**

Resources

- **WS-Reliable Messaging**

  http://www.ibm.com/developerworks/webservices/library/ws-rm/

  http://www.ibm.com/developerworks/webservices/library/ws-rmimp/

- **WS-Reliability**

  http://www.oasis-open.org/committees/download.php/6451/WS-Reliabil

Thank you!

chrisfer@us.ibm.com